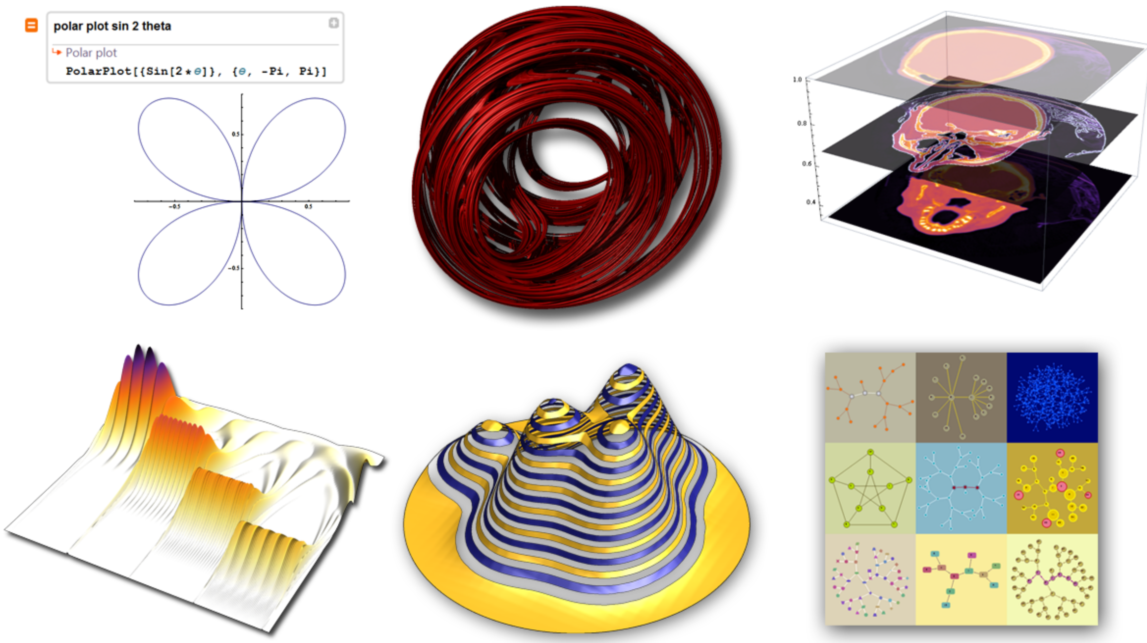
What's New in *Mathematica* 8

Wolfram *Seminar Series*

Presenter: *Paritosh Mokhasi*
seminars@wolfram.com
Jeff Todd: *jtodd@wolfram.com*
Justin Kehinde: *jkehinde@wolfram.com*

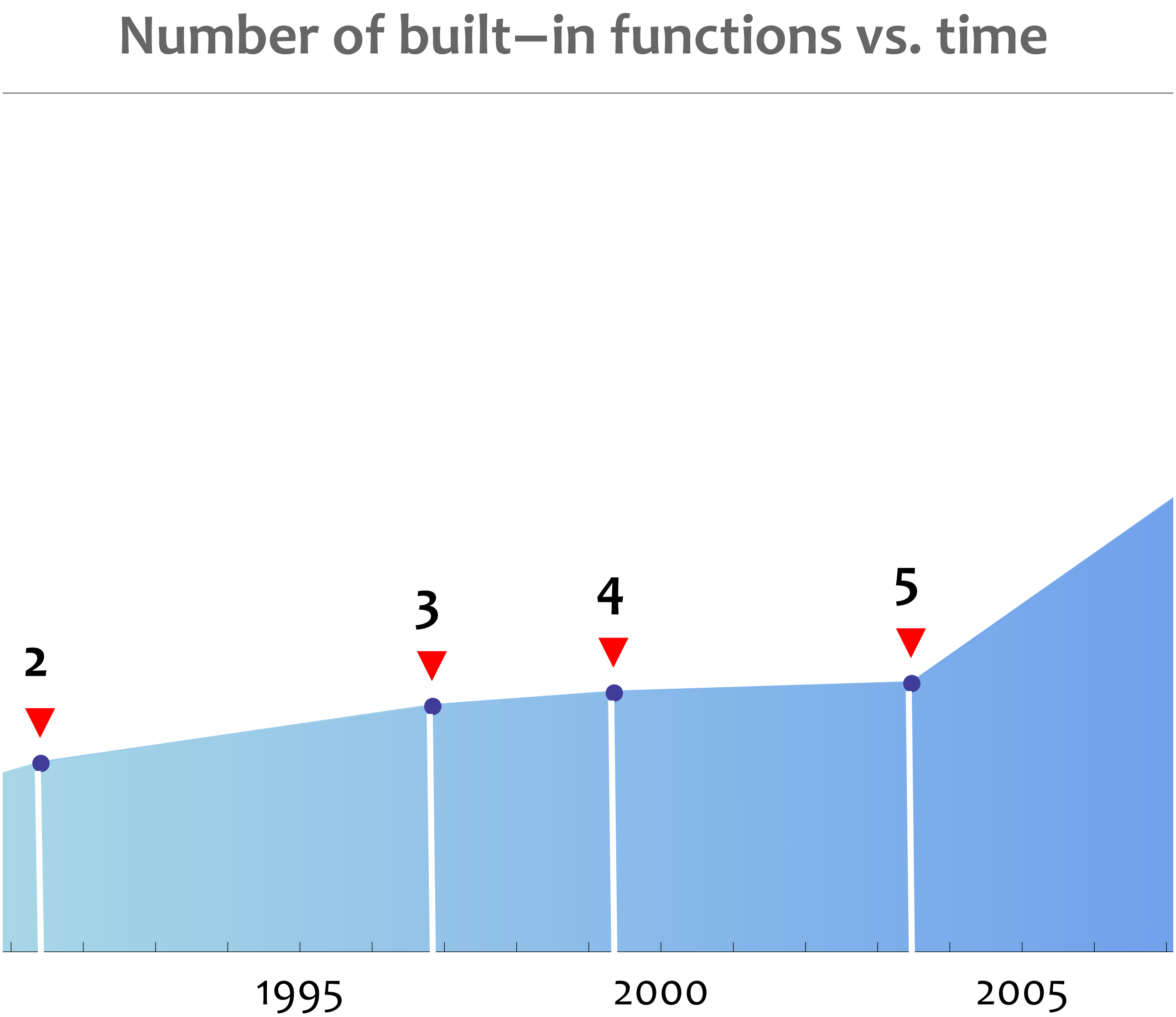
Mon 6 Jun 2011 12:13:32 (GMT –5.)

Introducing *Mathematica* 8



Mathematica 8 pioneers free-form linguistic input, introduces advanced capabilities in several application areas, and makes many improvements and additions to the world's most powerful mathematical, computational, and technical visualization platform.

Number of Built-In Functions



WHAT’S NEW

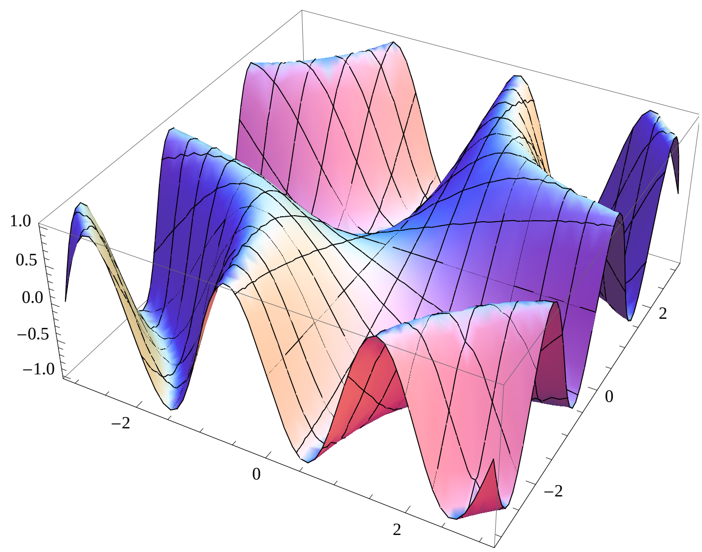
Wolfram|Alpha Integration

Free-Form Input and Accessing Trillions of Data

plot sin xy

3D plot

```
Plot3D[Sin[x * y],
  {x, -3.07, 3.07}, {y, -3.07, 3.07}]
```



Mathematica 8 integrates web-based access to the Wolfram|Alpha computational knowledge engine, making its vast repositories of data, computational knowledge, and linguistic interface available as part of *Mathematica*.

» **Free-form input**: get the precise *Mathematica* interpretation

» **Integration:** use *Mathematica* variables in free-form queries

» Access Wolfram|Alpha's huge repository of **computable data**

Free-Form Input: Examples

Free-form input allows you to use more of *Mathematica* without training, and teaches you *Mathematica* syntax.

Generate a plot using free-form input:

plot x sin x in red with short dashing

Customize the plot:

set line thickness 3

change background color to yellow

invert colors

For potentially dangerous commands, *Mathematica* just gives you a template:

export as a jpeg

Free-Form Input: Integration

Use *Mathematica* variables in free-form queries.

```
txt = ExampleData[{"Text", "OriginOfSpecies"}];
```

Mathematica asks if it can use information about your variables to generate better results:

strlen txt »

Input interpretation

```
StringLength[txt]
```

count occurrences of "Beagle" in txt »

Input interpretation

```
StringCount[txt, "Beagle"]
```

Wolfram|Alpha Data

Access all of Wolfram|Alpha’s data ...

... about cows:

cows in france since 1990

Or, about anything:

bucky ball 3d structure

Get the structured data for your own programmatic use:

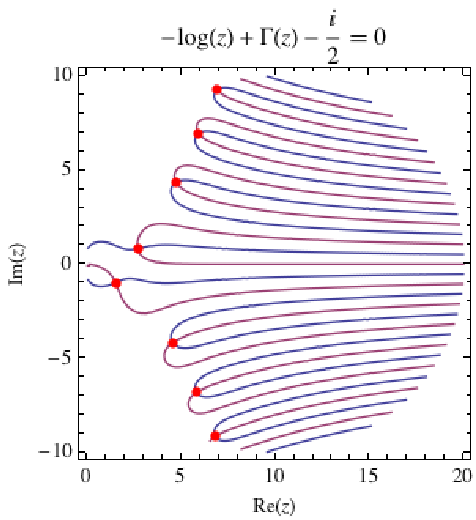
```
data = WolframAlpha["cows in UK", "DataRules"][[2, 2]]
DateListPlot[data, Joined → True, PlotRange → {0, All}]
```

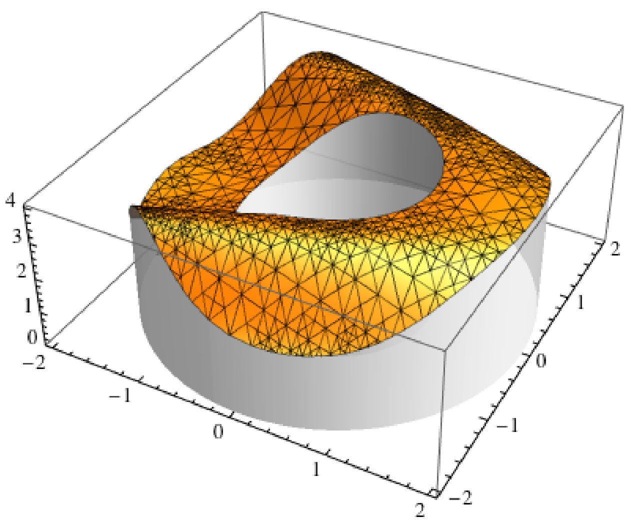
New Feature Summary

- » Free-form linguistic input
- » Full Wolfram|Alpha query inside *Mathematica*
- » Inline free-form input inside *Mathematica* code
- » WolframAlpha: programmatic access to the Wolfram|Alpha API

Core Algorithms

New and Improved





Mathematica 8 includes many improvements to its powerful core of symbolic and numerical algorithms.

» Fast **integer linear algebra**

» New methods for a very general class of **highly oscillatory numerical integrals**

» **Solve** over domains (reals, etc.), new **transcendental** and **high-degree polynomial** methods

» New **special functions** including probability and statistics functions

» Permutation group theory

Algorithms Enhancements

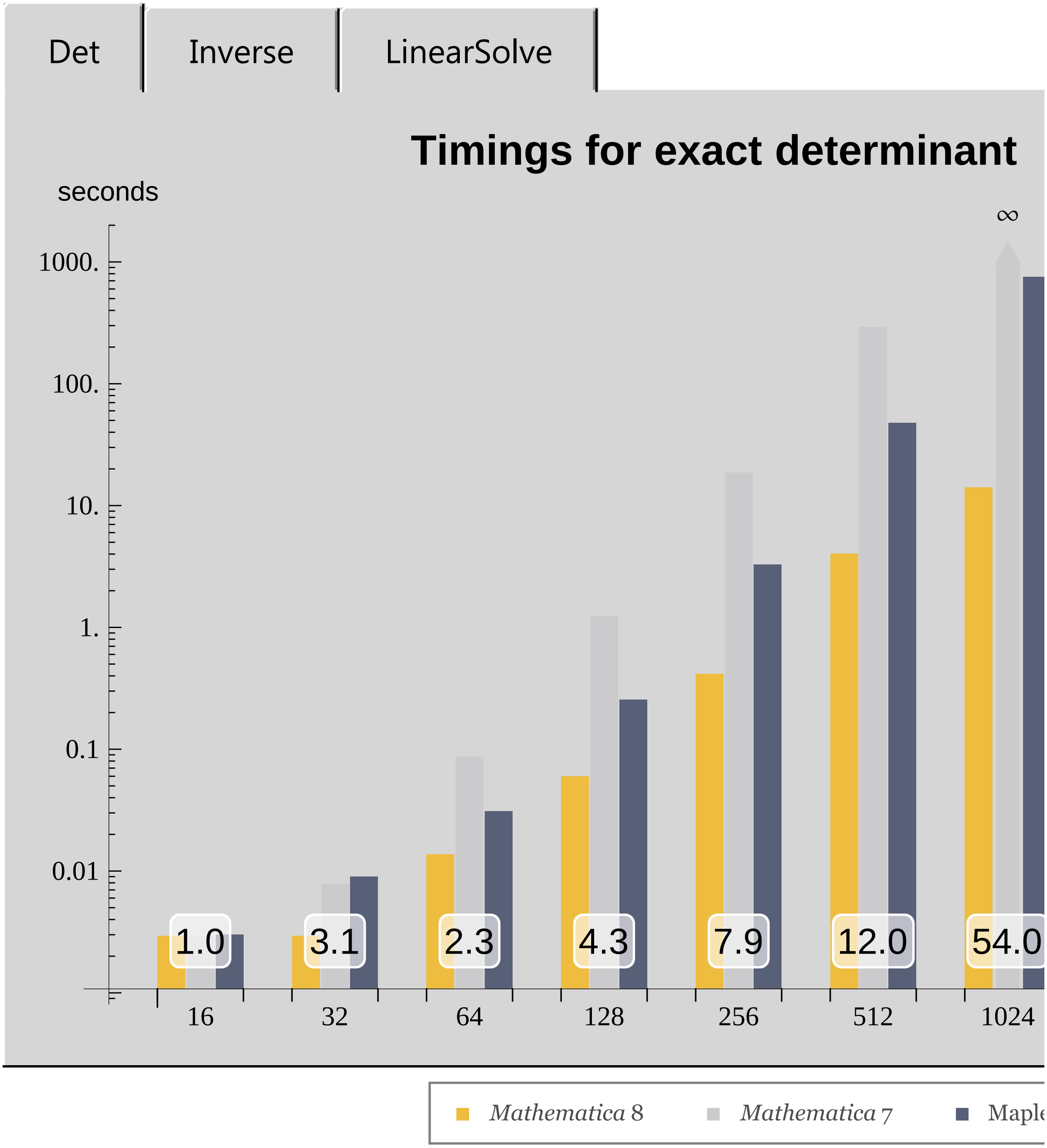
» Conditional results

Solve an equation over the reals and get a conditional result:

```
First[Solve[x^2 + y^2 == z^2 && x + y + 3 z == 1, {x, y}, Reals]]
```

» Fast exact linear algebra

Fast exact integer and rational linear algebra:



▲ Note: y-axis is shown in log scale.

» High oscillatory integration

SIAM challenge problem:

```
Plot[1/x Cos[Log[x]/x], {x, 0, 1},
 PlotRange → 15, Filling → Axis]
```

Integration

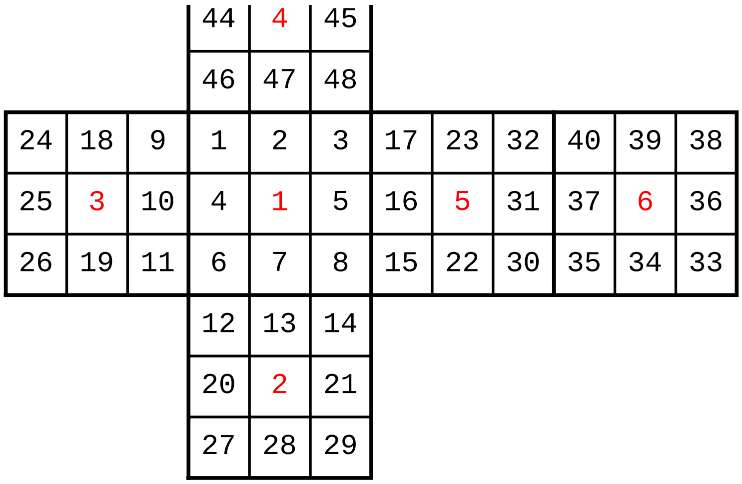
New methods handle highly oscillatory integration problems automatically:

```
NIntegrate[1/x Cos[Log[x]/x], {x, 0, 1}]
```

Built-in Group Theory

Mathematica 8 introduces a collection of algorithms and data structures for working with permutation groups.

The moves of a Rubik's cube form a permutation group:



These are six basic rotations:

```
rot1 = Cycles[{{1, 3, 8, 6}, {2, 5, 7, 4},
    {9, 48, 15, 12}, {10, 47, 16, 13}, {11, 46, 17, 14}}];
rot2 = Cycles[{{6, 15, 35, 26}, {7, 22, 34, 19},
    {8, 30, 33, 11}, {12, 14, 29, 27}, {13, 21, 28, 20}}];
rot3 = Cycles[{{1, 12, 33, 41}, {4, 20, 36, 44},
    {6, 27, 38, 46}, {9, 11, 26, 24}, {10, 19, 25, 18}}];
rot4 = Cycles[{{1, 24, 40, 17}, {2, 18, 39, 23},
    {3, 9, 38, 32}, {41, 43, 48, 46}, {42, 45, 47, 44}}];
rot5 = Cycles[{{3, 43, 35, 14}, {5, 45, 37, 21},
    {8, 48, 40, 29}, {15, 17, 32, 30}, {16, 23, 31, 22}}];
rot6 = Cycles[{{24, 27, 30, 43}, {25, 28, 31, 42},
    {26, 29, 32, 41}, {33, 35, 40, 38}, {34, 37, 39, 36}}];
```

Generate the group of rotations:

```
RubikGroup =
  PermutationGroup[{rot1, rot2, rot3, rot4, rot5, rot6}];
```

How many arrangements of a Rubik's Cube are there?

```
GroupOrder[RubikGroup]
```

Is it possible, by any sequence of moves, to swap positions 2 and 47?

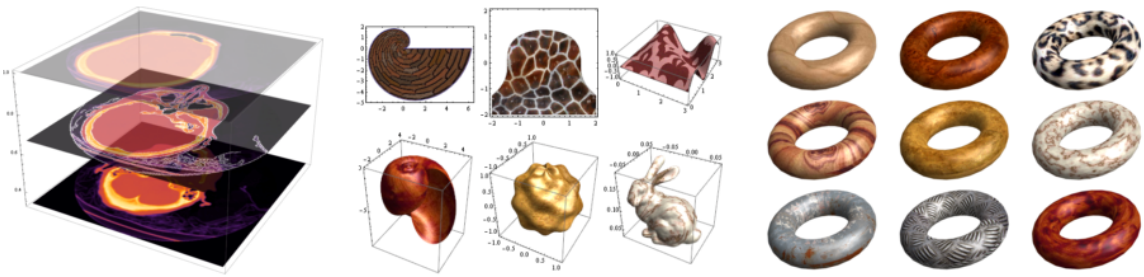
```
GroupElementQ[RubikGroup, Cycles[{{2, 47}}]]
```

What about simultaneously swapping two pairs of edge positions?

```
GroupElementQ[RubikGroup, Cycles[{{2, 47}, {31, 37}}]]
```

2D and 3D Graphics

Enhanced and Accelerated



Mathematica 8 is the leading choice for technical visualization, with several new features and performance enhancements for 2D and 3D graphics.

» Hardware accelerated **textures**

» New 2D graphics primitives: **filled curves** and **joined curves**

» Efficient support for multiple **geometric transformations**

Texture Everywhere

» Create new kinds of plots

Begin with a stream plot:

```
arrows = Rasterize[StreamPlot[
   Evaluate[{Re[(x + ⅈ y)^4 - 1], -Im[(x + ⅈ y)^4 - 1]}],
   {x, 0, 3}, {y, 0, 3}, VectorScale →
    {Automatic, Automatic, Log[#5 + 1] &}, Frame → False]]
```

Layer it onto a 3D plot:

```
Plot3D[Sin[x y], {x, 0, 3}, {y, 0, 3},
 Mesh → None, PlotStyle → Texture[arrows]]
```

» Textures can be anything

Use the current web cam image to texture a polygon:

Code

» Dynamic textures

Code

Curve Constructors: Joined and Filled

Mathematica 8 supports spline and piecewise linear curves including filled curves.

Extract outline curves from texts:

```
curve = First[First[ImportString[ExportString[Style["M8",
        FontFamily → "Times", FontSize → 72], "PDF"]]]] /.
   {Thickness[_] :→ {}, FilledCurve[args__] :→
     {FaceForm[ColorData["HTML", "Crimson"]],
      FilledCurve[args], Dashed,
      Arrow[JoinedCurve[args, CurveClosed → True]]}};
```

Use curves with arrows:

```
Animate[
 Graphics[{Arrowheads[{{0.05, t}, {0.05, Mod[t + 1/3, 1]},
      {0.05, Mod[t + 2/3, 1]}}], curve},
  ImageSize → Medium], {t, 0, 1}]
```

Multiple Geometric Transformations

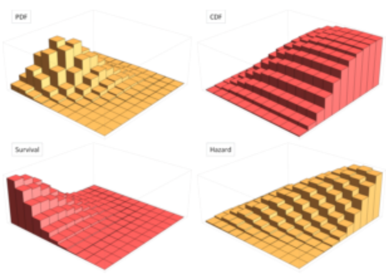
Mathematica 8 provides an efficient way to apply many geometric transformations to a single object.

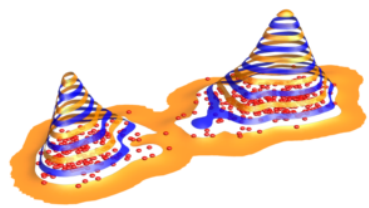
Create a textural image:

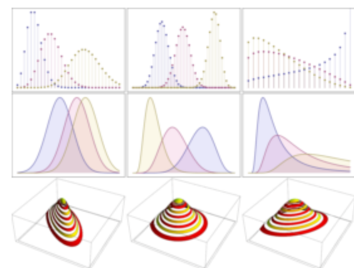
Code

Probability and Statistics

More Than Any Other System







The most complete coverage of statistical distributions and distribution properties of any system.

» Automatic **probability** and **expectation** calculations

» Most comprehensive set of built-in **parametric distributions**

» **Nonparametric distributions** defined from other distributions, formulas, or data

» **Distribution properties**: Moments, cumulant, random variate generation, ...

» **Statistics visualization**: quantile plot, box whisker plot, ...

Automatic Probabilities and Expectations

Sums and integrals with appropriate ranges and assumptions are constructed automatically behind the scenes.

A game involves players independently counting cars until they see a black car. What is the expected score of the winner of the game, if there are 3 players and 10% of cars are black?

```
Expectation[x, x ≈
  OrderDistribution[{GeometricDistribution[0.1], 3}, 3]]
```

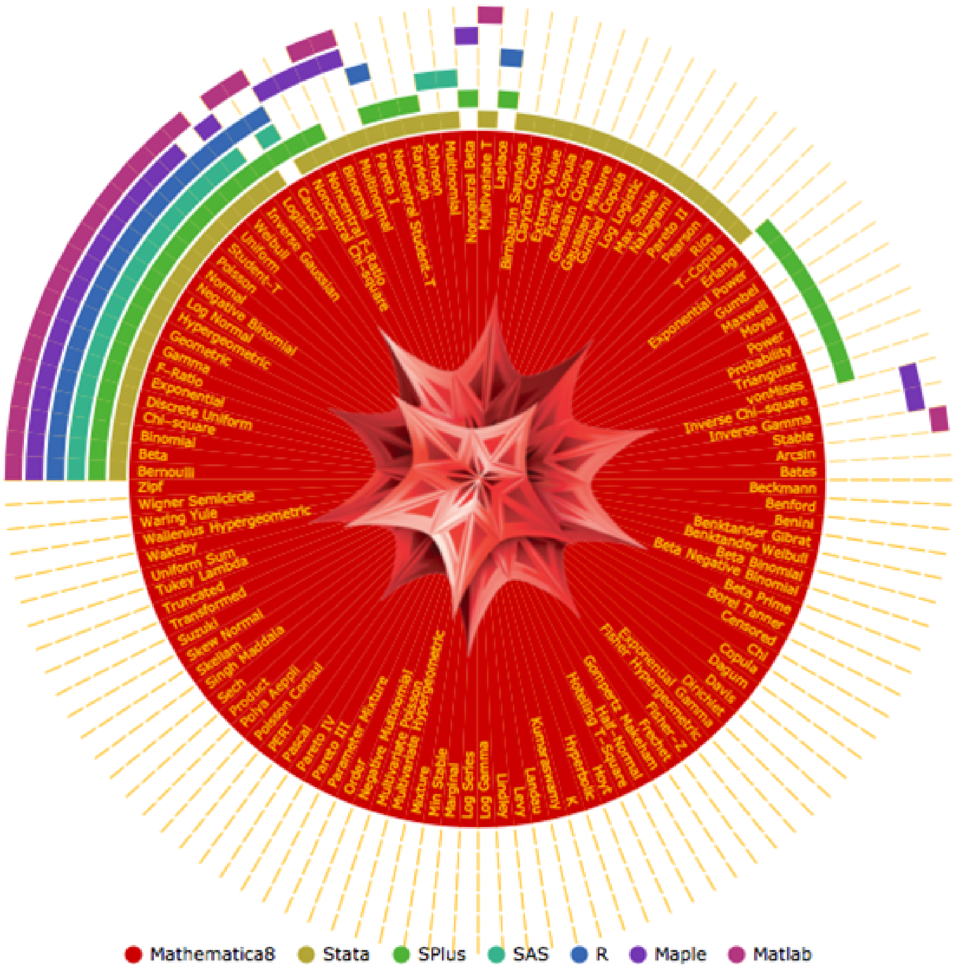
Automation

Chance that the winner counts fewer than 10 cars, given she counts at least 4?

```
Probability[Conditioned[x < 10, x ≥ 4], x ≈
  OrderDistribution[{GeometricDistribution[0.1], 3}, 3]]
```

Parametric Distributions

A vast range of built-in parametric distributions. (Click the image for distribution palette.)

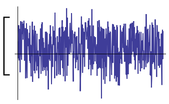
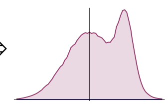


Nonparametric Distributions

» Distributions defined in terms of other distributions.

» Empirical distributions from data

Distributions defined from data.

SmoothKernelDistribution[ , …] ⟹ 

Inches of snowfall each year after 1910 in Buffalo:

```
dist = SmoothKernelDistribution[
  ExampleData[{"Statistics", "BuffaloSnow"}]]
```

Chance of more than 100 inches of snow:

```
Probability[x > 100, x ≈ dist]
```

» Automatic fitting of distributions to data

Distribution fitting is accomplished automatically, typically with methods like maximum likelihood.

Automation Fit stock prices to a lognormal distribution:

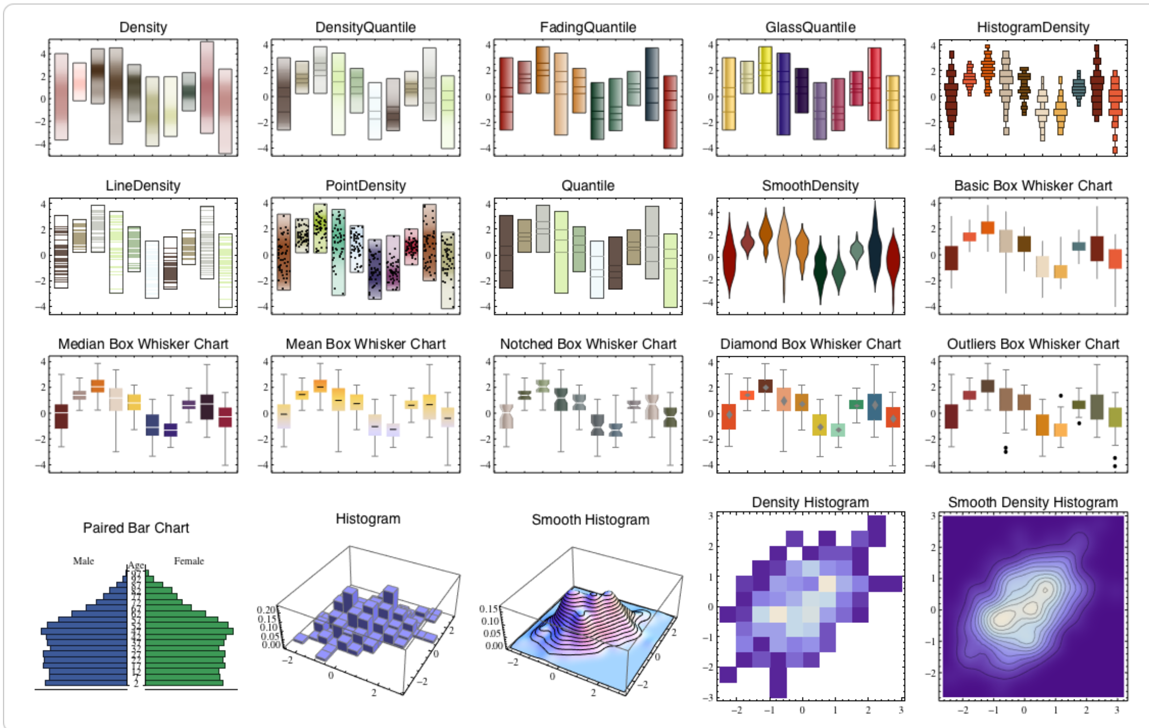
```
googleStock = FinancialData["GOOG",
   {{2006, 3, 10}, {2010, 3, 15}, "Day"}, "Value"];
dist = EstimatedDistribution[
  googleStock, LogNormalDistribution[μ, σ]]
```

Good fit—except at high quantiles!

```
QuantilePlot[googleStock, dist,
 Filling → Automatic, FillingStyle → Red]
```

Probability & Statistics Visualization

Data distribution charts, paired bar charts, histograms, and rectangle charts, and automatic probability scaling for existing histograms.



New Feature Summary

» Guide: Probability and Statistics »

» Many new parametric distributions: discrete and continuous, univariate and multivariate

» Derived distributions defined in terms of other distributions

» Nonparametric distributions defined from formulas or data

» Automatically calculate Probability and Expectation.

» Many new distribution properties:

PDF, CDF, SurvivalFunction, Moment, EstimatedDistribution, ...

» Random number generation (RandomVariate) from any distribution.

» Visualization: DiscretePlot3D, Histogram, PairedHistogram, QuantilePlot, ProbabilityPlot, BoxWhiskerChart, DistributionChart, ...

Financial Engineering

Compute, Develop, and Visualize

In addition to *Mathematica*'s core capabilities, which make it a natural platform for developing financial engineering solutions, *Mathematica* 8 adds specific valuation capabilities for many kinds of financial instruments.

» **Time value calculations** for annuity, interest, cash flow-based securities

» **Value, implied volatility, and Greeks** calculations

» Huge range of derivatives and bonds

Indicators and Time Values

» Built-in Financial Indicators

Absolute Price Oscillator	Acceleration Bands	Accumulation Distribution Line	Accumulative Swing Index
Aroon	Aroon Oscillator	Average Directional Movement Index	Average Directional Movement Index Rating
Average True Range	Bollinger Bands	Chaikin Money Flow	Chaikin Oscillator
Chaikin Volatility	Chande Momentum Oscillator	Close	Close Location Value
Close Location Value Volume	Commodity Channel Index	Commodity Selection Index	Demand Index
Detrended Price Oscillator	Directional Movement	Double Exponential Moving Average	Dynamic Momentum Index
Ease Of Movement	Exponential Moving Average	Fast Stochastic	Force Index
Forecast Oscillator	Full Stochastic	High	Highest High
Inertia	Intraday Momentum Index	Keltner Channels	Klinger Oscillator
Linear Regression Indicator	Linear Regression Slope	Linear Regression Trendlines	Low
Lowest Low	Market Facilitation	Mass Index	Median Price
M E S A Phase	M E S A Sine Wave	Momentum	Money Flow Index
Moving Average Convergence Divergence	Moving Average Envelopes	Negative Volume Index	On Balance Volume
Open	Parabolic Stop And Reversal	Percentage Price Oscillator	Percentage Volume Oscillator
Performance	Polarized Fractal Efficiency	Positive Volume Index	Price Channels
Price Volume Trend	Projection Bands	Projection Oscillator	Q Stick
Raff Regression Channel	Random Walk Index	Range Indicator	Rate Of Change
Relative Momentum Index	Relative Strength Index	Relative Volatility Index	R Squared
Simple Moving Average	Slow Stochastic	Standard Deviation	Standard Deviation Channels
Standard Error	Standard Error Bands	Stochastic Momentum Index	Swing Index
Time Segmented Volume	Time Series Forecast	Triangular Moving Average	Triple Exponential Moving Average
T R I X	True Range	Typical Price	Ultimate Oscillator
Variable Moving Average	Vertical Horizontal Filter	Volatility System	Volume
Volume Rate Of Change	Weighted Close	Weighted Moving Average	Wilders Moving Average

» Cashflows, Interest, and Annuities

Symbolic representation of cash flows, effective interest rates, annuities, etc.

Time value today of $1 paid later, assuming continuously compounded interest rate *r*:

```
TimeValue[1, EffectiveInterest[r, 0], -t]
```

Combine with new probability and statistics features:

Expected value of $1 death benefit paid at time *t*, where *t* is drawn from a Gompertz-Makeham distribution:

```
nsp = Expectation[%, t ≈ GompertzMakehamDistribution[a, b]]
```

Integration

Find the premium, paid yearly in advance, that is necessary to make the expected present value of that payment stream equal to the net single death benefit:

```
Solve[
 Expectation[TimeValue[
    AnnuityDue[premium, t], EffectiveInterest[r, 0], 0],
   t ≈ GompertzMakehamDistribution[a, b]]
  == nsp, premium]
```

This is an important equation in life insurance, now automatically derivable in *Mathematica*.

Built-in Financial Derivatives

» Superfunction for financial derivatives

Automation Compute time value of a derivative with ambient financial parameters:

```
FinancialDerivative[
 {"American", "Call"},
 {"StrikePrice" → 40, "Expiration" → 1},
 {"CurrentPrice" → 30, "Dividend" → 0.05,
  "Volatility" → 0.3, "InterestRate" → 0.05},
 {"Value", "CriticalValue"}]
```

» Comprehensive coverage

List of built-in financial derivatives:

Altiplano	American Call	American Put	Annapurna
Asian Arithmetic European Call	Asian Arithmetic European Put	Asian Geometric European Call	Asian Geometric European Put
Atlas	Barrier Down In American Call	Barrier Down In American Put	Barrier Down In European Call
Barrier Down In European Put	Barrier Down Out American Call	Barrier Down Out American Put	Barrier Down Out European Call
Barrier Down Out European Put	Barrier Up In American Call	Barrier Up In American Put	Barrier Up In European Call
Barrier Up In European Put	Barrier Up Out American Call	Barrier Up Out American Put	Barrier Up Out European Call
Barrier Up Out European Put	Binary Asset European Call	Binary Asset European Put	Binary Cash European Call
Binary Cash European Put	Chooser European	Compound Call European Call	Compound Call European Put
Compound Put European Call	Compound Put European Put	Double Barrier Knock In American Call	Double Barrier Knock In American Put

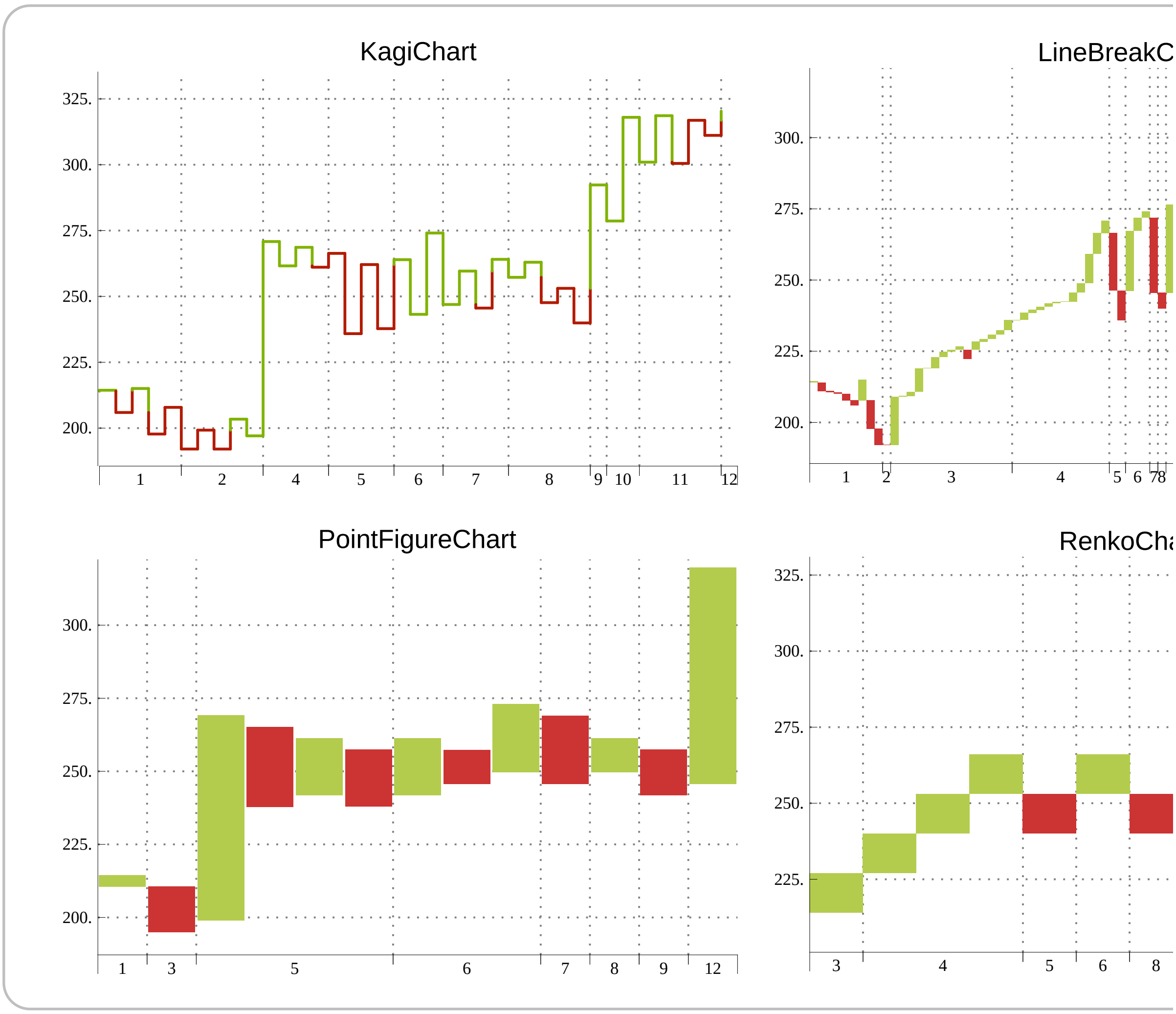
Double Barrier Knock In European Call	Double Barrier Knock In European Put	Double Barrier Knock Out American Call	Double Barrier Knock Out American Put
Double Barrier Knock Out European Call	Double Barrier Knock Out European Put	European Call	European Put
Everest	Extendible Holder European Call	Extendible Holder European Put	Extendible Writer European Call
Extendible Writer European Put	Future European Call	Future European Put	Himalaya
Lookback Fixed European Call	Lookback Fixed European Put	Lookback Floating American Call	Lookback Floating American Put
Lookback Floating European Call	Lookback Floating European Put	One Touch American Call	One Touch American Put
Perpetual American Call	Perpetual American Put	Perpetual Lookback Call	Perpetual Lookback Put
Quanto Fixed Exchange American Call	Quanto Fixed Exchange American Put	Quanto Fixed Exchange European Call	Quanto Fixed Exchange European Put
Quanto Fixed Strike American Call	Quanto Fixed Strike American Put	Quanto Fixed Strike European Call	Quanto Fixed Strike European Put
Rainbow Best American	Rainbow Best European	Rainbow Max American Call	Rainbow Max American Put
Rainbow Max European Call	Rainbow Max European Put	Rainbow Min American Call	Rainbow Min American Put
Rainbow Min European Call	Rainbow Min European Put	Rainbow Money American	Rainbow Money European
Rainbow Worst American	Rainbow Worst European	Russian	Spread American Call

Spread American Put	Spread European Call	Spread European Put	Vanilla American Call
Vanilla American Put	Vanilla European Call	Vanilla European Put	

Financial Engineering Visualization

Many types of specialized financial charts.

Selection of charts plotting Apple share price changes over 2010, used in, e.g., technical/trend analysis:



Interactive trading charts showing price with automated financial indicators:

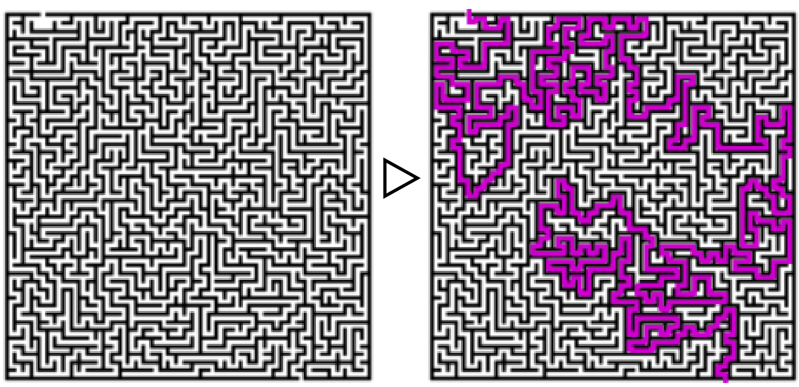
```
InteractiveTradingChart[
 {"AAPL", {{2010, 1, 1}, {2010, 12, 31}}}]
```

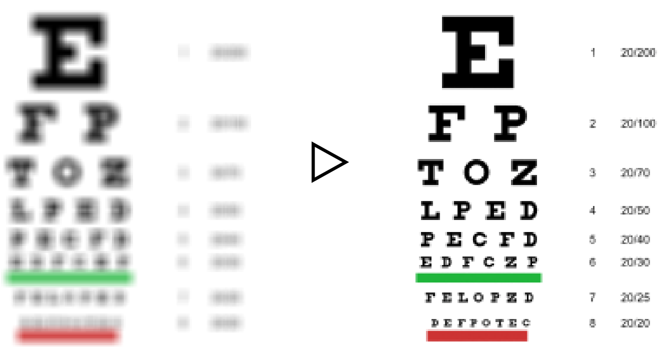
New Feature Summary

- Guide: Finance »
- Valuation of financial derivatives, and implied volatility and Greeks calculations
- Valuation of financial bonds
- Huge range of financial indicators
- Time value of money calculations for annuity, interest, and cashflow securities
- Many new financial visualizations including interactive trading charts

Image Processing

Comprehensive Image Processing Environment





Mathematica's modern approach to general image processing, started in *Mathematica* 7, has been massively extended in *Mathematica* 8, making it the most powerful platform for image processing algorithm development.

» Real-time image acquisition: Direct webcam input support

» Morphology, segmentation, and feature detection: components, skeletons, edge and corner detection, ...

» Arbitrary geometric transformations and image alignment

» Video import: QuickTime, import/export individual frames, etc.

» Text recognition, inpainting, smoothing, noise removal, etc.

Live Image Capture

Access image acquisition devices such as webcams dynamically.

Apply a filter to the current image from the webcam:

```
EdgeDetect[CurrentImage[]]
```

Integration Tyranno-vision (seeing only moving objects):

```
Dynamic[ImageDifference @@ CurrentImage[2]]
```

Broken mirror:

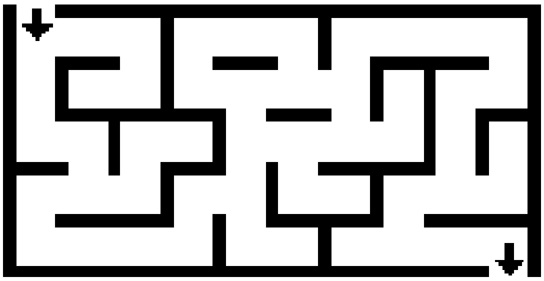
```
DynamicModule[
 {img1, img2, size = 15, pos = {0, 0}, dx = 320, dy = 240},
 Dynamic[img1 = ImagePad[CurrentImage[],
    {{dx, dx}, {dy, dy}}, "Reflected"];
  img2 = ImageTake[img1, dy {1, 2} - pos[[2]],
    (dx {1, 2}) - pos[[1]]];
  ImageAssemble[Reverse /@
    ImagePartition[img2, {size, dy}]]]]
```

Morphological Operations

Work with image skeletons.

Simplify a maze by thinning it:

```
Thinning[
```



```
]
```

Find solutions by pruning the skeleton:

```
Pruning[%, ∞]
```

Integration

Express the maze as a graph:

```
MorphologicalGraph[%, VertexCoordinates → None]
```

Geometric Transformations

» Perspective transformations

Perspective transformations allow you to effectively view an image from another position and orientation:

A famous painting by Holbein, “The Ambassadors”. Can you find the hidden human skull?

```
img = 
```

The skull is visible from an extreme perspective (Wolfram blog post):

```
ImagePerspectiveTransformation[img,
  TransformationFunction[( 0.5   -0.87  -0.05
                          -1.34   2.7    0.43
                          -0.62   0.2    0.525 )]]
```

Integration

Use the painting as the Texture of a Polygon in a Graphics3D and Animate it:

Code

» Geometric transformations with any functions

Any pixel-wise transformation can be specified.

Transform the Mona Lisa to produce a wave effect:

```
Manipulate[ImageTransformation[
```

```
  ,
  {First[#1] + 1/10 Sin[p Last[#1]], Last[#1]} &,
  Padding → "Reflected"], {p, 1, 30}]
```

Feature Detection

Detect edges, lines, and key points.

Find prominent lines in a photograph:

img = 

```
Show[img, Graphics[{Thick, Orange, Line /@
    ImageLines[Binarize[GradientFilter[img, 1]], .2]}]]
```

Find and visualize matching points in two different images of the moon:

imgs = { ,  };

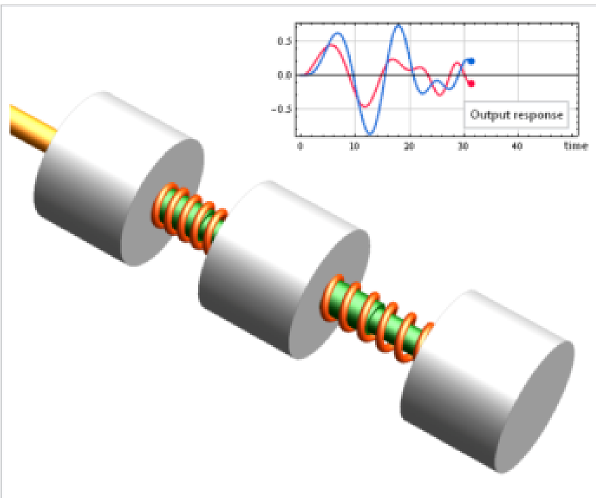
```
matches = ImageCorrespondingPoints @@ imgs;
MapThread[Show[#1, Graphics[{Yellow, MapIndexed[
      Inset[#2[[1]], #1] & , #2]}]] &, {imgs, matches}]
```

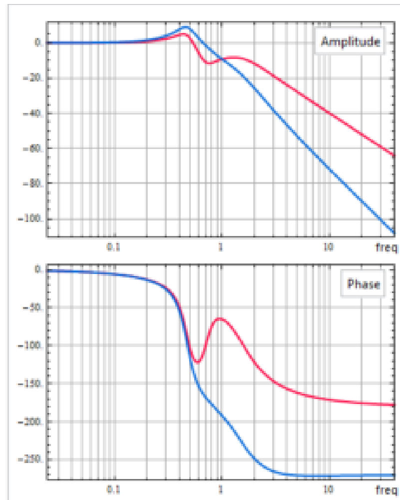
New Feature Summary

- Guide: Image Processing & Analysis »
- New morphological operations: Thinning, Pruning, ...
- Component analysis: ComponentMeasurements, SelectComponents, ...
- Image segmentation: MorphologicalBinarize, ImageForestingComponents, ClusteringComponents, WatershedComponents, ...
- Feature and key point detection: EdgeDetect, ImageLines, CrossingDetect, CornerFilter, ImageKeypoints, TextRecognize, ...
- Geometric transformations and image alignment: ImageAlign, ImageCorrespondingPoints, ImageTransformation, ...
- New image filters and deconvolution: WienerFilter, MeanShiftFilter, TotalVariationFilter, ImageDeconvolve, ...
- QuickTime and other video format support, real-time capture from imaging devices

Control Systems

Integrated System for Design and Analysis





Mathematica 8 includes a comprehensive library of functions for the design, analysis, and simulation of control systems.

- **Symbolic representation** of state space and transfer function models, in continuous or discrete time
- Define model by ODEs, with **automatic linearization**
- **Join systems** (in series, in parallel, extract subsystems, ...)
- **Visualization**: Bode plots, Nyquist plots, Nichols plots, ...
- **Optimal control** algorithms

Example: Coupled Damped Oscillators

Mathematica introduces a unifying symbolic representation of transfer function and state space models.

Get a state space model from differential equations for coupled damped oscillators:

```
model = StateSpaceModel[
  {2 x2[t] - x3[t] + (3 x2'[t])/2 - (x3'[t])/2 + x2''[t] == F[t],
   -x2[t] + x3[t] - (x2′[t])/2 + (x3′[t])/2 + 2 x3″[t] == 0},
  {{x2[t], 0}, {x2′[t], 0}, {x3[t], 0}, {x3′[t], 0}},
  {{F[t], 0}}, {x2[t], x3[t]}, t]
```

Convert between state space representation and classical transfer function representation:

```
TransferFunctionModel[model]
```

Immediately ask questions about the system.

Can the model be fully controlled with the specified input F[t]?

```
ControllableModelQ[model]
```

Automation Simulate the model with varying driving frequency:

```
Plot[Evaluate[OutputResponse[model,
   Sin[2^(t/15)], {t, 0, 100}]], {t, 0, 100}]
```

Analyze the model:

Plot frequency response gain for amplitude and phase:

```
BodePlot[model, GridLines → Automatic,
 ImageSize → Medium, GridLinesStyle → GrayLevel[2/3]]
```

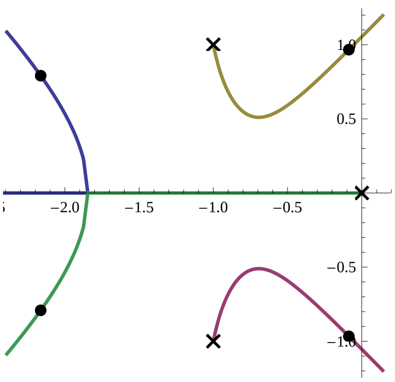
Combine with *Mathematica*'s 3D graphics, animate, and export:

```
Animate[Evaluate[
   ShowGeom[1, OutputResponse[model, 1 / 2, {t, 0, 50}], t]],
  {t, 0, 50, 0.1}, AnimationRunning → False,
  DefaultDuration → 20]
```

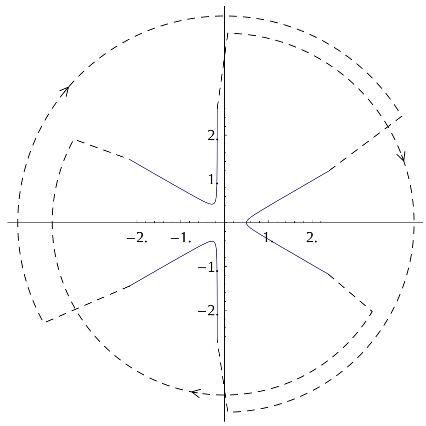
Control Systems Visualization

Many standard plots, especially important in classical analysis.

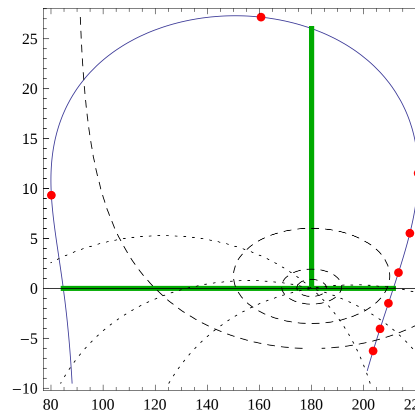
RootLocusPlot



NicholsPlot



NyquistPlot

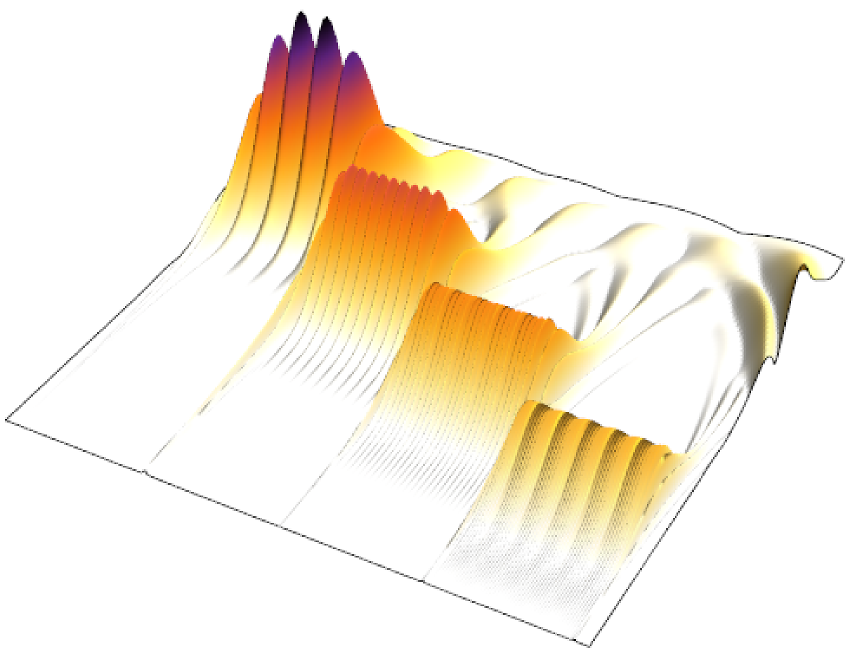


New Feature Summary

- Guide: Control Systems »
- Continuous or discrete-time StateSpaceModel and TransferFunctionModel, including conversion between them, and automatic construction by linearizing ODEs
- Join systems in different ways: series, parallel, feedback, ...
- Analysis and design: GainPhaseMargins, controllability and observability, StateFeedbackGains, LQRegulatorGains, KalmanEstimator, ...
- Model simulation: StateResponse, OutputResponse
- Visualization: BodePlot, NyquistPlot, NicholsPlot, RootLocusPlot, SingularValuePlotv, ...

Wavelet Analysis

Fastest, Most Comprehensive Wavelet Tool



Mathematica 8 now includes the fastest, most comprehensive library of wavelet analysis tools anywhere.

» **Discrete** and **continuous** wavelet transforms with all standard **wavelet families**

» Operate on arrays of **any dimension** and **any precision**, and directly on image and sound data

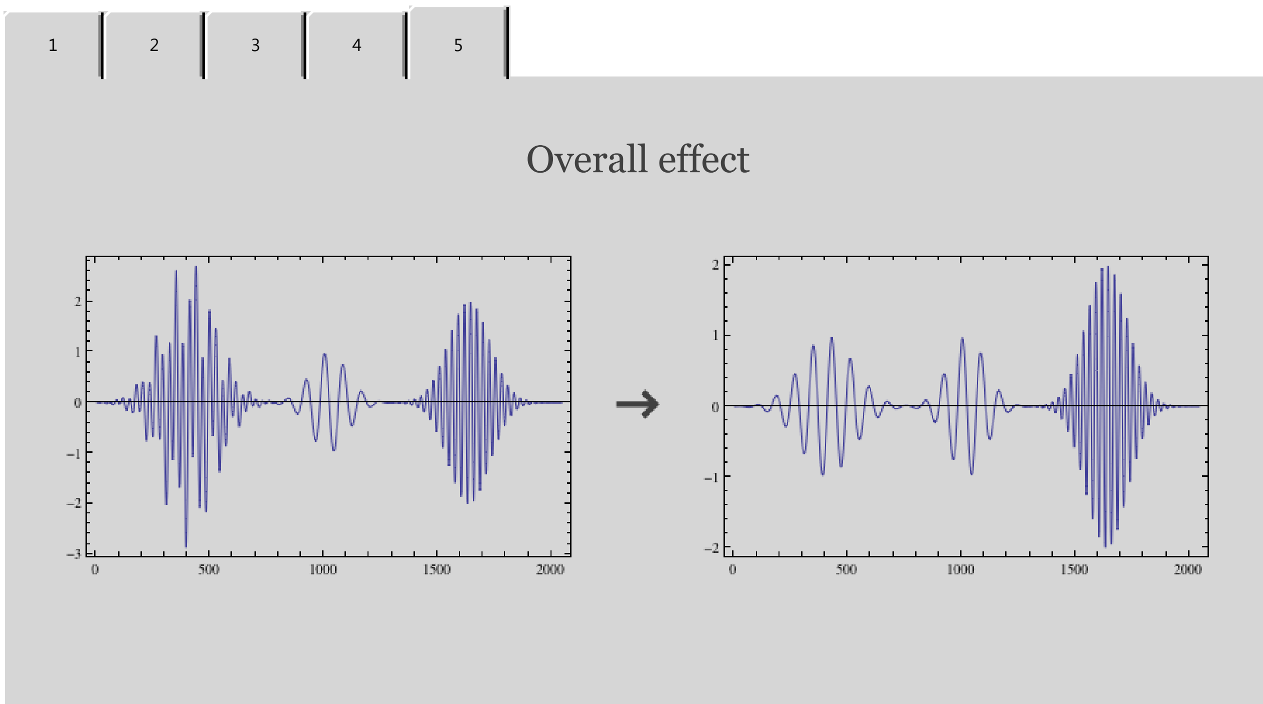
» **Symbolic representation** of transform coefficients

» Wavelet **algorithms** and **visualization**

What are Wavelet Transforms Used for?

Wavelets localize noise and features in frequency and time simultaneously, allowing more selective filtering and analysis:

Remove one of two overlapping signals:



Different Data, One Wavelet Tool

Transforms work directly on images, sound, or tensor data.

Wavelet transform of image:

dwd = DiscreteWaveletTransform[ , Automatic, 3]

The coefficients are also images. Visualize in a pyramid layout:

```
WaveletImagePlot[dwd]
```

Here's something impossible in any other system:

4-dimensional, 70 digit precision wavelet transform:

```
dwd = DiscreteWaveletTransform[
  RandomReal[1, {6, 6, 6, 6}, WorkingPrecision → 70],
  WorkingPrecision → 70]
```

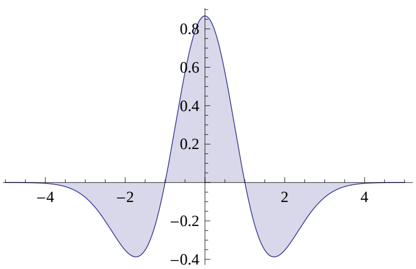
View one of the coefficients:

```
dwd[{0, 0, 4}]
```

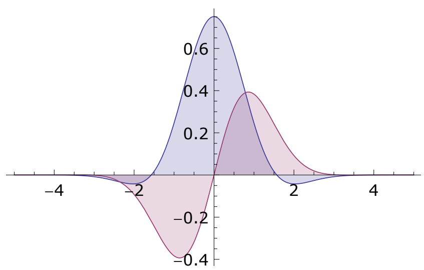
Wavelet Families

All standard wavelets for discrete and continuous wavelet transforms are supported:

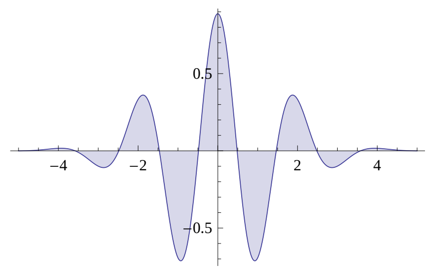
MexicanHatWavelet



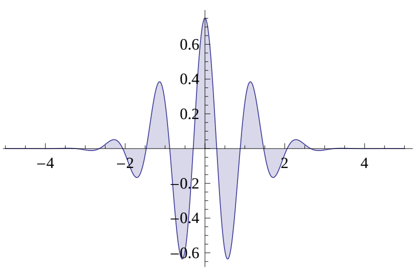
GaborWavelet



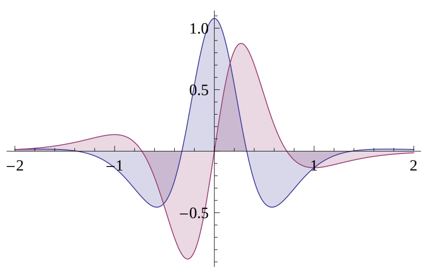
DGaussianWavelet



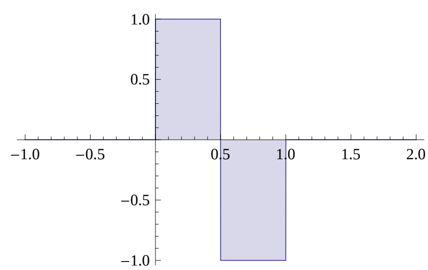
MorletWavelet



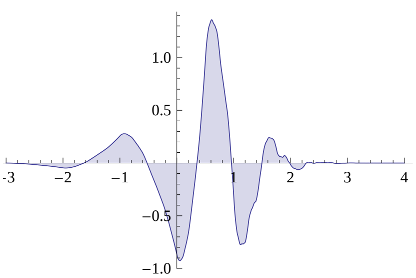
PaulWavelet



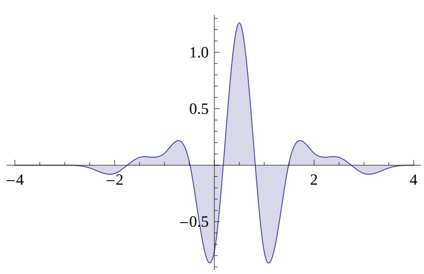
HaarWavelet



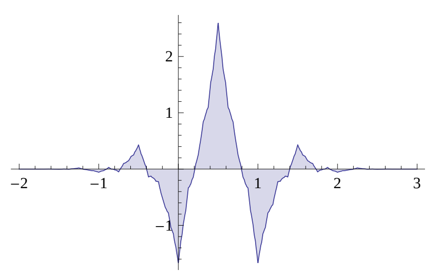
DaubechiesWavelet



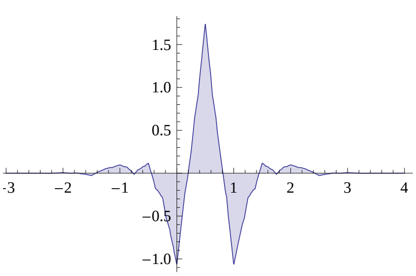
BattleLemarieWavelet



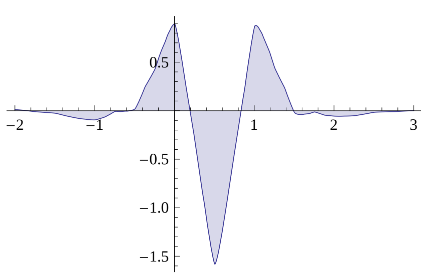
BiorthogonalSplineWavelet



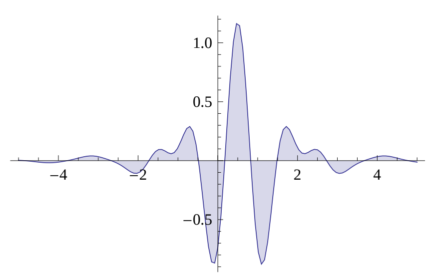
CDFWavelet



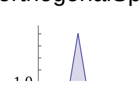
CoifletWavelet



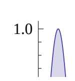
MeyerWavelet



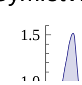
ReverseBiorthogonalSplineWavelet

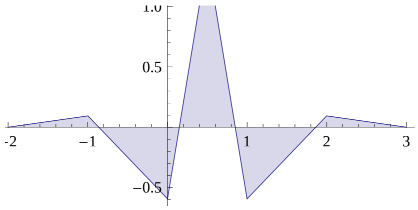


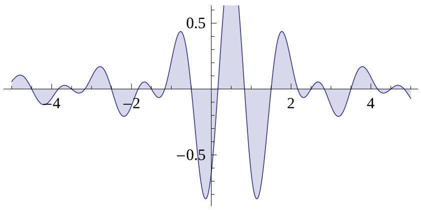
ShannonWavelet

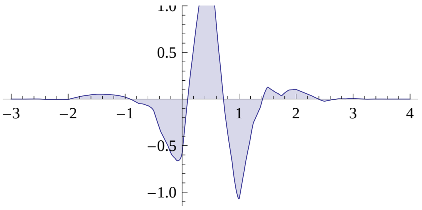


SymletWavelet




1.0
0.5
-0.5
-2
-1
1
2
3

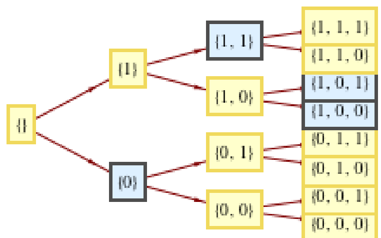

0.5
-0.5
-4
-2
2
4


1.0
0.5
-0.5
-1.0
-3
-2
-1
1
2
3
4

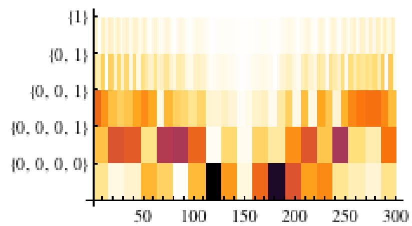
Wavelets Visualization

Many different ways to visualize value and structure of wavelet coefficients:

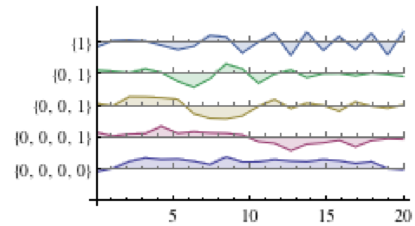
Plot coefficient tree



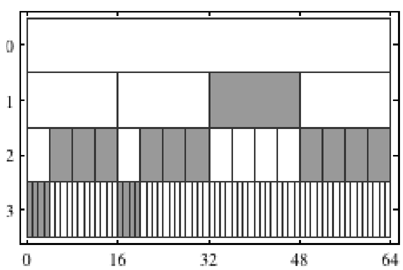
Discrete scalogram



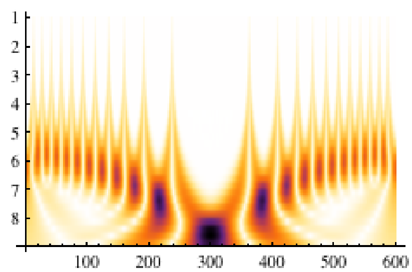
Plot 1–D wavelet coefficients



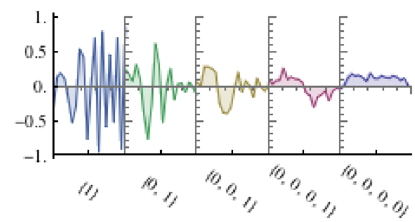
Block view of coefficient tree



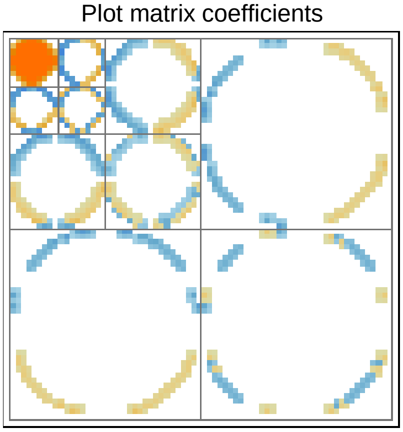
Continuous scalogram



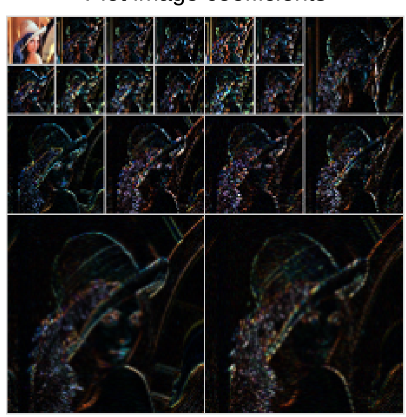
Plot 1–D wavelet coefficients
(alternate layout)



Plot matrix coefficients



Plot image coefficients

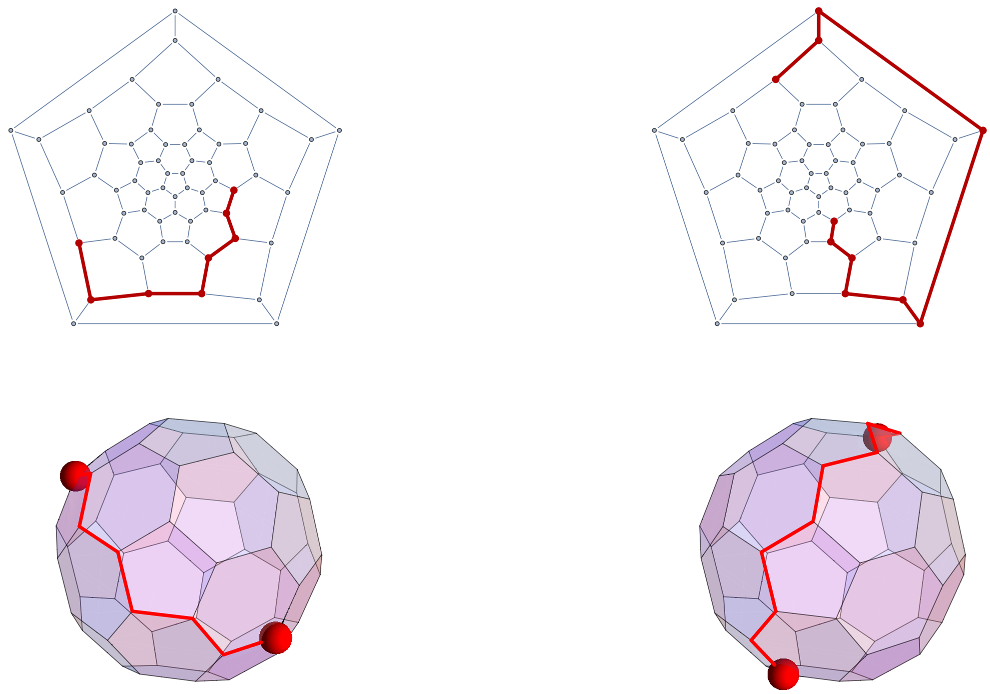


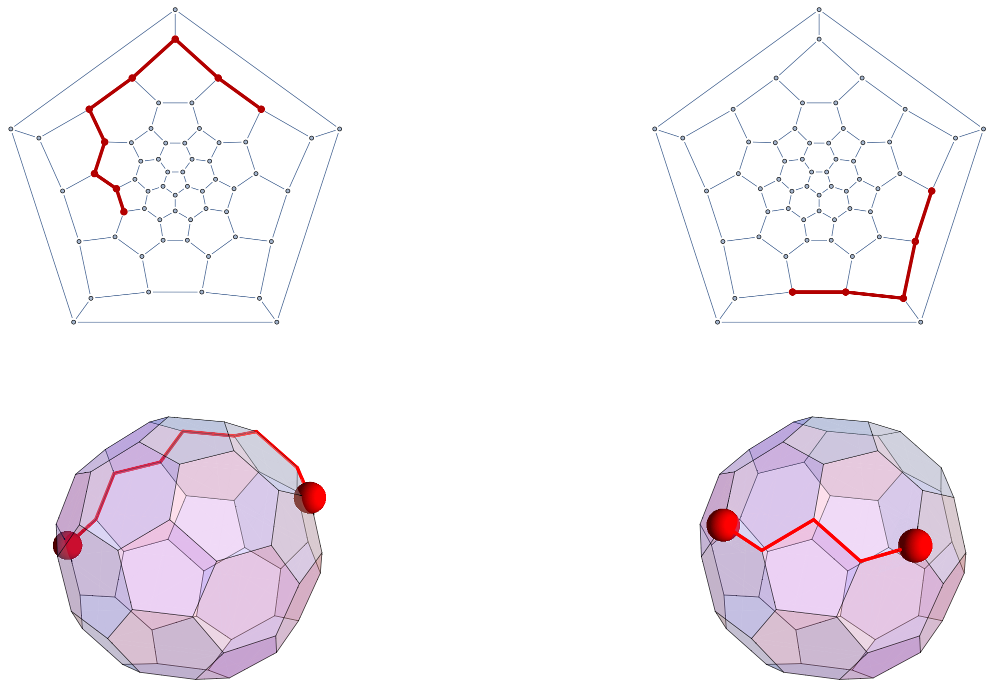
New Feature Summary

- Guide: Wavelet Analysis »
- Forward and inverse discrete transforms: DiscreteWaveletTransform and StationaryWaveletTransform
- Packet transforms: DiscreteWaveletPacketTransform and StationaryWaveletPacketTransform
- LiftingWaveletTransform including integer lifting
- ContinuousWaveletTransform and inverse, including exact inverse transform
- Large library of standard wavelet families:
  - Continuous transforms (MexicanHatWavelet, GaborWavelet, ...)
  - Discrete transforms (HaarWavelet, DaubechiesWavelet, ...)
- Wavelet properties: scaling function WaveletPhi, wavelet function WaveletPsi, ...
- Symbolic representations: DiscreteWaveletData and ContinuousWaveletData
- Wavelet algorithms: WaveletThreshold and WaveletBestBasis
- Visualization: WaveletScalogram, WaveletListPlot, WaveletImagePlot, WaveletMatrixPlot

Graphs & Networks

Integrated and Scalable: For Modeling and Analysis





Mathematica 8 introduces state-of-the-art functionality for modeling, analyzing, synthesizing, and visualizing graphs and networks.

» Fully integrated **symbolic representation** of graph and network modeling

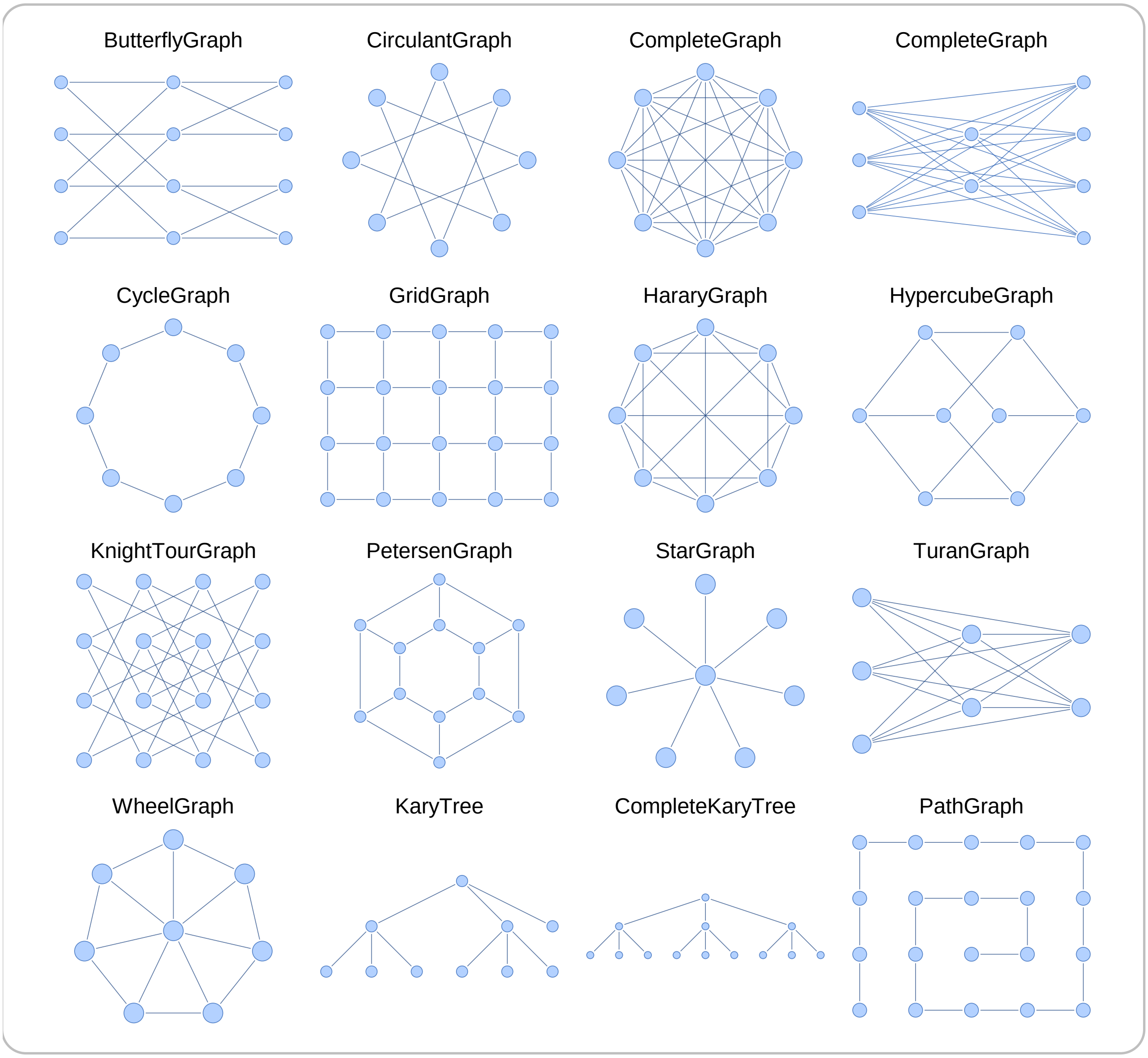
» **Graph operations:** Union, difference, power, etc.

» State-of-the-art **analysis algorithms:** Find covers, isomorphisms, paths, cycles, connected components, etc.

» **Import & export:** Many new graph formats

Graphs & Networks: Built-in Graph Families

Mathematica 8 has an extensive collection of special families of graphs:



Graph & Network Analysis

» Find subgraphs

Highlight largest cliques (complete subgraphs) in random graphs:

```
Grid@Table[HighlightGraph[g = RandomGraph[{8, 16}],
   Subgraph[g, First[FindClique[g]]]], {2}, {4}]
```

» Find cycles

The edges and vertices of a dodecahedron form a graph:

```
{Graphics3D[{Opacity[.8],
   PolyhedronData["Dodecahedron", "Faces"]}],
 g = PolyhedronData["Dodecahedron", "SkeletonGraph"]}
```

Solve Hamilton's "*Icosian Game*": find a closed Hamiltonian cycle (visiting every vertex):

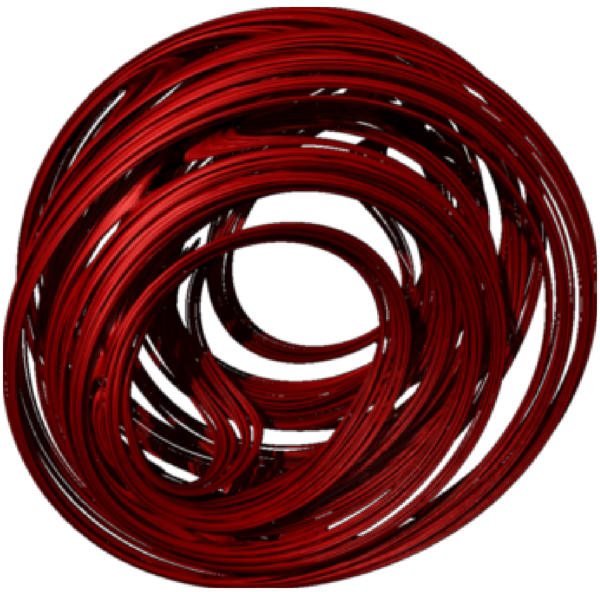
```
h = PathGraph@First[FindHamiltonianCycle[g]];
{HighlightGraph[g, h, GraphHighlightStyle → "Thick"],
 Graphics3D[{Opacity[.8],
   PolyhedronData["Dodecahedron", "Faces"], Red, Tube[
    PolyhedronData["Dodecahedron", "VertexCoordinates"][[
     Append[VertexList[h], VertexList[h][[1]]]]], 0.1]}]}
```

New Feature Summary

- Guide: Graphs and Networks »
- Graph and network modeling:
  - Parametric families of graphs (CompleteGraph, WheelGraph, ...)
  - Random graphs (RandomGraph)
  - Graphs from morphological decomposition of images (MorphologicalGraph)
- Import & export popular graph formats: "GraphML", "GXL", ...
- Extensive styling, labeling, and layout options for edges and vertices
- Graph operations: Subgraph, NeighborhoodGraph, adding and removing edges and vertices, union and difference, ...
- Graph measures and metrics: BetweennessCentrality, PageRankCentrality, distance measures, ...
- Algorithms: paths, cycles, connected components, cliques and covers, ...

Programming and Development

C Integration, Parallelized, Dynamic Library, and GPU Power



Mathematica 8 introduces C code generation, compilation, and linking, including automatically parallelized natively compiled function objects.

» **Generate** C code and **compile** to a dynamic library, or to a standalone executable

» **Link** dynamic libraries into *Mathematica* at run-time

» **Automatically** do all of the above, through Compile, and run compiled functions in **parallel**

» Program for **GPUs** in CUDA or OpenCL

C Code Generation

Generate optimized C code. *Mathematica* includes all the headers and libraries to build standalone executables from this code.

```
s = ExportString[Compile[x, Sin[x^2]], "C"];
```

```
#include "math.h"

#include "WolframRTL.h"

static WolframCompileLibrary_Functions funStructCompile;

static mbool initialize = 1;

#include "m0000221901.h"

DLLEXPORT int Initialize_m0000221901(WolframLibraryData libData)
{
if( initialize)
{
funStructCompile = libData->compileLibraryFunctions;
initialize = 0;
```

Compilation & Parallelization

In *Mathematica* 8 the Compile function can automatically generate C code, compile it, link dynamically to *Mathematica* at run-time, and run in parallel (multi-threading) when possible.

Create a compiled function that runs in native code and in parallel:

```
f = Compile[{{c, _Complex}},
  Module[{num = 1}, FixedPoint[(num++; #1^2 + c) &, 0,
    99, SameTest → (Re[#1]^2 + Im[#1]^2 ≥ 4 &)]; num],
  CompilationTarget → "C", RuntimeAttributes → {Listable},
  Parallelization → True]
```

Easily plot 200000 evaluations of the function:

```
ArrayPlot[
 f[Table[x + ⅈ y, {x, -2, 1/2, 0.005}, {y, -1, 1, 0.005}]],
 ColorFunction → "Rainbow"]
```

Explore the Mandelbrot set interactively using the natively compiled function:

Code

GPU Programming Support

Build GPU computation into your *Mathematica* programs using new support for CUDA and OpenCL environments.

CUDA Link package

```
<< CUDALink`
CUDAInformation[]
?CUDA*
```

CUDAImageConvolve[,

```
{{1, 1, 1}, {1, -8, 1}, {1, 1, 1}}]
```

» CUDALink Guide »

OpenCL Link package

```
<< OpenCLLink`
OpenCLInformation[]
OpenCLFractalRender3D[]
```

» OpenCLLink Guide »

New Feature Summary

» Guide: C/C++ Language Interface »

» Updated Compile function:

Compile to make listable, parallel, and native code

(Requires C/C++ compiler on each platform, mostly freely available)

» GPU computing using CUDALink and OpenCLLink

(CUDALink requires NVIDIA's graphics cards)

» C Code Generator: Export *Mathematica* expressions and compiled functions to C code in C99 standard

» Wolfram *Library Link*: Load functions from dynamic libraries

» C Compiler Driver: Automatically operate any standard C compiler

» Symbolic C: Fully symbolic representation of C code

Summary

There are many more features than we have time for in one seminar—see the full list of new features.

If you have not yet given the latest version of *Mathematica* a test drive, be sure to check the box provided in the seminar survey, indicating you want to download a free trial version of *Mathematica* 8.

» Further Resources

» Courses and Webinars »

» Learning Center »

» Interactive Documentation »

» Wolfram Support Center »

» What's New in 8 »

Initialization

» Color

» C Compiler

» Special Cells

» Spring in 3D